IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 29, 2020, accepted June 7, 2020, date of publication July 13, 2020, date of current version July 22, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3008854

Variational Bayesian Group-Level Sparsification for Knowledge Distillation

YUE MING[1], (Member, IEEE), HAO FU[1], YIBO JIANG[2], AND HUI YU[3], (Senior Member, IEEE)

[1]Beijing Key Laboratory of Work Safety and Intelligent Monitoring, School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China
[2]China Ningbo XiTang Information Technologies, Inc., Ningbo 315500, China
[3]School of Creative Technologies, University of Portsmouth, Portsmouth PO1 2UP, U.K.

Corresponding author: Yue Ming (myname35875235@126.com)

This work was partly supported by Beijing Natural Science Foundation of China (No. L182033), and partly by Fundamental Research Funds for Central Universities (No. 2019PTB-001).

ABSTRACT Deep neural networks are capable of learning powerful representation, but often limited by heavy network architectures and high computational cost. Knowledge distillation (KD) is one of the effective ways to perform model compression and inference acceleration. But the final student models remain parameter redundancy. To tackle these issues, we propose a novel approach, called Variational Bayesian Group-level Sparsification for Knowledge Distillation (VBGS-KD), to distill a large teacher network into a small and sparse student network while preserving accuracy. We impose the sparsity-inducing prior on the groups of parameters in the student model, and introduce the variational Bayesian approximation to learn structural sparseness, which can effectively prune most part of weights. The prune threshold is learned during training without extra fine-tuning. The proposed method can learn the robust student networks that have achieved satisifying accuracy and compact sizes compared with the state-of-the-arts methods. We have validated our method on the MNIST and CIFAR-10 datasets, observing 90.3% sparsity with 0.19% accuracy boosting in MNIST. Extensive experiments on the CIFAR-10 dataset demonstrate the efficiency of the proposed approach.

INDEX TERMS Knowledge distillation, group sparsity, sparsity-inducing prior, variational Bayesian approximation.

I. INTRODUCTION

Deep learning (DL) has shown impressive power in various applications, such as computer vision, speech recognition and natural language processing [1]. However, the current deep learning networks usually suffer from high computational expense and memory consumption, which hinder their deployment in resource-limited devices in real-world scenarios. For example, VGG-16 [2] has more than 100 million parameters and require more than 19.6 billion floating-point operations (FLOPs) when processing an image with a 224×224 resolution. Therefore, it is critical to reduce the memory consumption and accelerate inference for the cumbersome models.

Recently, model compression techniques have emerged with tremendous progress, which can be roughly categorized into four categories: low-rank approximation [3], parameter quantization and sharing [4], network sparsification and pruning [5], and knowledge distillation (KD) [6], [7]. Among these approaches, KD has been proved to obtain considerable compression ratio while making less damage to accuracy than other methods [8]. It encourages the students to mimic the teacher's behavior by minishing the additive loss term, which measures the distance between the student model and the teacher model. However, the performance of KD is very sensitive to knowledge definition and transmission mechanism. Many existing efforts [9] have been taken to modify knowledge metrics to get more information from teachers. But few have studied the sparseness in the final student models.

In this paper, we propose a novel method, called Variational Bayesian Group-level Sparsification for Knowledge Distillation (VBGS-KD), which can further compress student model by effectively leveraging the benefit of network sparsification and pruning from a Bayesian point of view. First, we construct a standard pipeline for KD, and innovatively impose sparsity inducing priors on groups of parameters to capture the nature structural correlation of neural network connections. Then, we perform Bayesian Inference for forcing the approximate

The associate editor coordinating the review of this manuscript and approving it for publication was Ziyan Wu.

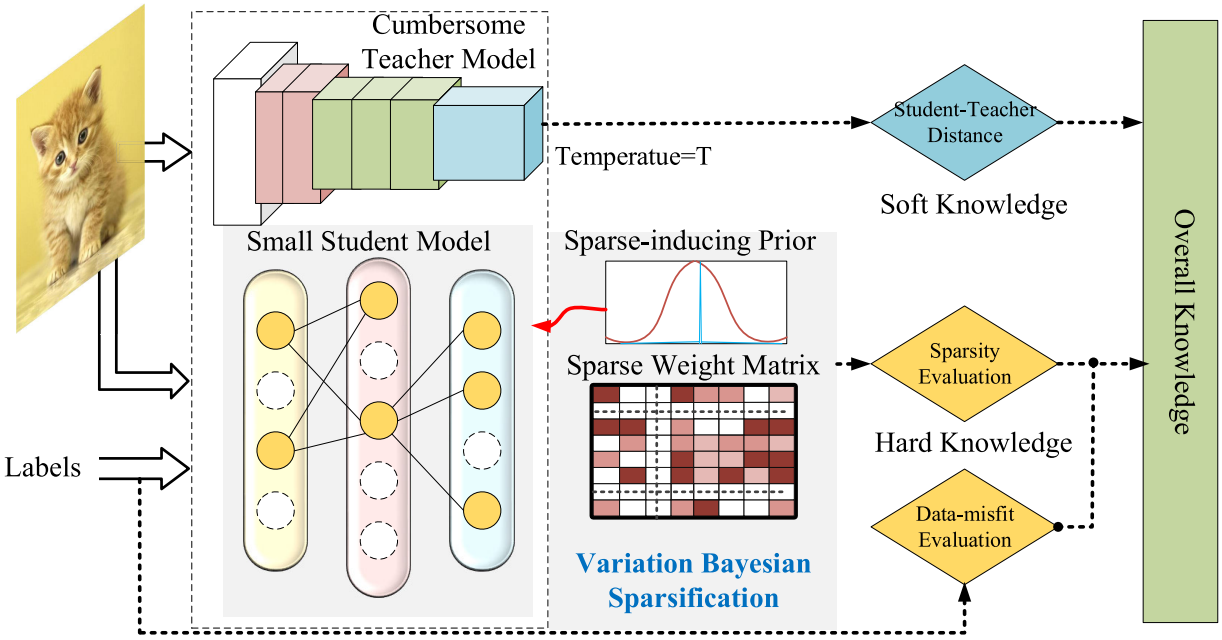


FIGURE 1. Framework of our Variational Bayesian Group-level Sparsification for Knowledge Distillation (VBGS-KD). The standard distillation process is denoted in the dotted box. Two knowledge flows, soft knowledge and hard knowledge, are combined to construct overall knowledge. Variational Baysian sparsification is performed to further compress the student model.

posterior distribution close to the sparse prior, which can effectively learn the structure sparseness of the student models during training. Finally, we prune the redundant parameters in group according to the learned-threshold to obtain the final compact model. The flow chart of the proposed method is shown in Figure 1.

The main contributions of the proposed method are summarized as follows:

1) **Bayesian Sparsification for Knowledge Distillation.** Our Bayesian sparsification in KD learns the best network structure during the training process and infers the optimal dropout rates with high accuracy and fine-tune free.
2) **Variational Group-level Sparsification.** We further expand the sparsification into group-level by introducing the sparse-inducing prior for group of parameters instead of individual weights, which can effectively describe the intrinsic correlation (dependency) of the model weight tensors. The randomness of the interval estimation makes the result more accurate and the estimation is more reliable.
3) **State-of-the-art Compression Result.** The proposed method has achieved superior performance on different datasets, including MNIST and CIFAR-10. Our method can simultaneously realize the accuracy and compression ratio boosting by knowledge distillation and complexity reduction from variational sparsification.

The rest of this paper is organized as follows. We survey related works on model compression in Section 2. Our proposed framework is described in Section 3. The following Section 4 demonstrates the experimental evaluations and results. Finally, we conclude the paper in Section 5.

II. RELATED WORK

We discuss the model compression methods closely related to our research, which can be roughly categorized as: low-rank approximation, quantization and weight sharing, network sparsification and pruning, and knowledge distillation.

Low-rank approximation assumes that the filters or weights of CNN lie on a low-rank subspace, it utilizes matrix or tensor decomposition to reduce parameter dimensions and time complexity [10]. The typical Singular Value Decomposition (SVD) worked well on fully-connected layers and large convolutional kernels [11], but usually performed poorly for small kernels. Although these methods can achieve a relatively good compression rate but followed by an accuracy drop.

Quantization and weight sharing attempts to group weights with similar values to reduce the number of free parameters, including codebook-based quantization [12] and low-bit quantization [13]. The former ones constrained weights and hashed into different groups before training. The weights for each group were shared and only the shared weights and hash indices needed to be stored. However, they required pre-determined codebook and computational efficiency algorithm. The latter ones focused on using fixed-point data to represent the weight of CNN [13], including binary weight and ternary weight [14]. These work showed that neural networks could be trained using only low-bit fixed point format with little accuracy degradation. However, the codebook values were pre-determined and the

quantization group was handcrafted. It thus, caused quantization loss uncontrollable. Besides, they only partitioned weights based on the weight level and could not achieve a good result in extremely low-bit quantization.

Network sparsification and pruning reduces network complexity by removing the connections based on weight magnitudes [15] or significant scores [16]. The work in [17] exploited the sparsity in DNN and perform pruning. Han *et al.* [5] jointly learned weights and connections to remove the least important weights according to their absolute values. Similar sparsification and pruning schemes have been explored based on Bayesian viewpoint [18]. However, the unstructured sparse weight-matrices still need index for non-zero weights or have to be stored in special format, which does not fit well in parallel computation. Then, group-sparsity regularizer was proposed to prune neurons or convolutional kernels and obtained thinned dense matrices [19]. There are some other works combine both the weight and group sparsification for model compression or feature selection [34]. However, most prior works need extra fine-tuning to recovery its accuracy.

Knowledge distillation (KD) focuses on distillating the teacher model as ''knowledge'', introducing a general technique for knowledge transfering [20]. Guo *et al.* [21] improved the student network to produce more confident predictions with the help of the teacher network for robust student network learning. Li *et al.* [22] proposed a dynamic saliency estimation approach for aerial videos via spatiotemporal knowledge distillation, which could effectively remove the inter-model redundancy. Zhang *et al.* [23] learned to distill future knowledge from a backward neural language model (teacher) to future-aware vectors (student) during the training phase, which were incorporated into the attention layer to provide full-range context information. Yang *et al.* [24] proposed a fast scene text detector, where the teacher network guided the training process of a student via knowledge distilling for maintaining the tradeoff between accuracy and efficiency. However, these methods independently extracted the instance features from the single teacher. The prior works [25] determined the importance of parameters or filters based on the gradients of the trained model with the repetive trails and careful tuning. Ashok *et al.* [26] used reinforcement learning to guide pruning but required two-stage learning. Simultaneous distillation algorithms [27] were developed to simplify the distillation training process. Wang *et al.* [28] proposed collaborative learning for bidirectional model assistance, which is a flexible strategy to collect mutual information and provide assistance by using a mutual knowledge base (MKB). However, the above methods still leave parameter redundancy in the final student model.

III. VARIATIONAL BAYESIAN GROUP-SPARSIFICATION DISTILLATION

Knowledge distillation is an effective model compression technique, which serves well for our ultimate goal to obtain small and high performance student models. However, most existing methods do not pay attention to parameter redundancy in the student model. In this section, we first review the general knowledge distillation formulation. Then, we study the effect of group sparsification for the student model to demonstrate the compactness. Based on this finding, we further propose the variational group sparsity as an optimal learning problem as shown in Figure 1. Our proposed approach devotes to effectively learn the structure sparseness of the student model during training, and infers the optimal dropout rates from data, resulting high accuracy preserved and fine-tune free.

A. KNOWLEDGE DISTILLATION

In the knowledge distillation framework, there are two knowledge flows. One is obtained from teacher guide, called **soft knowledge** L_S. Another is from ground-truth labels termed **hard knowledge** L_H.

In the $\mathcal{C}$-category classification task, the hypothesis function estimating the probability $p(\hat{y} = c_i|\boldsymbol{x}, t; \boldsymbol{\theta})$ is calculated by a ''softmax'' function:

$$Q(\boldsymbol{x}, t) = p(\hat{y} = c_i|\boldsymbol{x}, t; \boldsymbol{\theta}) = \frac{\exp(z_i/t)}{\sum_{j=1}^{\mathcal{C}} \exp(z_j/t)}, \tag{1}$$

where $\boldsymbol{\theta}$ denotes all parameters of model, z_i is the logits, t is noted as temperature that is normally set to 1. It is found that a higher t produces a softer pseudo-probability distribution over classes [20], which provides implicit knowledge for student model learning.

1) SOFT KNOWLEDGE

To quantify the alignment between the student model and the teacher in their predictions, we introduce the Kullback-Leibler (KL) divergence and the soft loss is computed by:

$$L_S = t^2 KL(Q_S(\boldsymbol{x}, t), Q_T(\boldsymbol{x}, t)), \tag{2}$$

where Q_S, Q_T denote class probability of student model and teacher model, respectively. t^2 cancels out the $1/t^2$ term of the gradients.

2) HARD KNOWLEDGE

Hard knowledge in our algorithm, as shown in Eq. 3, consists of two parts: data-misfit evaluation and sparsity evaluation. In classic supervised learning, data-misfit between the output of student network $Q_S(\boldsymbol{x}, t)$ and the ground-truth label y is usually penalized by cross-entropy $\mathcal{CE}(\cdot)$ loss. And we introduce prior information by variational Bayesian approximation to impose sparsity constraint. Here, we denote the sparsity term as $\mathcal{R}(\boldsymbol{\theta})$, and will further discuss in the following Section.

$$L_H = \underbrace{\mathcal{CE}(Q_S(x, t), y)}_{\text{Data misfit evaluation, } L_H^1} + \underbrace{\mathcal{R}(\boldsymbol{\theta})}_{\text{Sparsity evaluation, } L_H^2} \tag{3}$$

3) OVERALL LOSS

Finally, we utilize a weighted average of these two kinds of knowledge, and obtain the overall loss function:

$$\mathcal{L} = \lambda L_S + (1-\lambda)L_H, \tag{4}$$

where λ is a hyper-parameter controlling the tradeoff between two loss functions.

B. GROUP SPARSENESS IN NETWORKS

For compression, we aim at enforcing model parameter θ to contain a small number of nonzero weights, while the majority of the parameters are zero. This goal is translated into the optimization problem using sparsity-promoting penalty function. The most common example is the L^1-norm based formulation, also known as LASSO. The optimization formulation for LASSO is

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \sum_n ||\boldsymbol{\theta}^\top x - y||_2^2 + \eta||\boldsymbol{\theta}||_1, \;\; ||\boldsymbol{\theta}||_1 = \sum_i \sum_j |\theta_{ij}|, \tag{5}$$

here, η is the regularization parameter controlling the proportion of the enforced sparsity.

In the traditional sparse modeling, the sparsity constraint is imposed on individual weights, θ_{ij}. Recently, a different modeling approach generalized traditional sparse methods, where sparsity is enforced on groups instead of the individual weights to obtain structural sparsity. The group-LASSO with G groups is settled by

$$\hat{\theta} = \arg\min_{\boldsymbol{\theta}} \sum_n ||\boldsymbol{\theta}^\top x - y||_2^2 + \eta||\boldsymbol{\theta}||_{1,\mathcal{G}}, \; ||\boldsymbol{\theta}||_{1,\mathcal{G}} = \sum_g ||\theta_g||_2. \tag{6}$$

The group sparse model achieves high compress ratio.

However, the aforementioned deterministic methods often give a point estimation of the model, but the variance of estimated parameter, a desirable property, is often ignored. Bayesian group-LASSO has emerged, which finds the solution and its variance to represent the confidence of the model in the estimation.

Here, we clarify the concept of groups in a network. For the fully connected layer, the group corresponds to a neuron, the input neuron corresponds to the row of weight matrices, and the output corresponds to the column. For convolutional layers, the group corresponds to a filter. Overall, we have a total of G groups, corresponding to three specific effects on the resulting network. If the variables of an input group are set to zero, the corresponding feature can be neglected during the prediction phase for effective feature selection. Then, if a variable in a hidden group is set to zero, we can remove the corresponding neuron for pruning effect and a thinner hidden layer. Finally, if a variable in a bias group is set to zero, we can remove the corresponding bias from the neuron. The groups of parameters are constructed as illustrated in Figure 2. Such regularization will bring several benefits, such as speeding up inference and improving generalization.

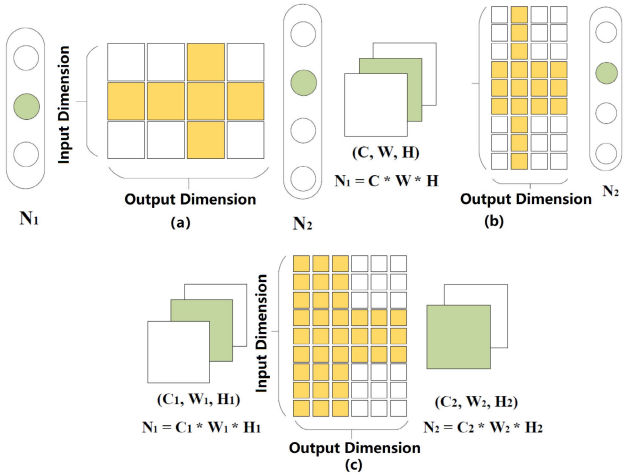


FIGURE 2. Group-level sparsity in network. All outgoing connections from a single neuron or filter (corresponding to a group) will be either simultaneously zero, or not. Neuron pruning results in (a) matrix compression between two full-connected layers, (b) filter pruning's effects on matrix between convolutional layer and full-connected layer, (c) filter prune between two convolutional layers. Colored blocks indicate pruned weights. Sparsity is enforced on groups instead of the individual element.

C. VARIATONAL BAYESIAN SPARSIFICATION IN STUDENT MODEL

Due to the structural dependencies among the weights of $\boldsymbol{\theta}$, we introduce an additional set of random variable to capture the dependencies and propose to extend the model as group sparsity. Group-level sparsification is achieved under variational Bayesian approximation.

The posterior distribution $p(\boldsymbol{\theta}|\mathcal{D})$ is approximated by a parametric distribution $q_\phi(\boldsymbol{\theta})$. This approximation's quality is measured by $D_{\mathrm{KL}}(q_\phi(\boldsymbol{\theta})\|p(\boldsymbol{\theta}|\mathcal{D}))$. Generally, KL minimization is transformed to *the Evidence Lower Bound (ELBO)* maximization. The optimal value of variational parameters ϕ can be found by maximization $L(\phi)$:

$$L_H = L(\phi) = \underbrace{\mathbb{E}_q(\log p(D|\boldsymbol{\theta}))}_{\text{Data misfit evaluation}} + \underbrace{-D_{\mathrm{KL}}(q_\phi(\boldsymbol{\theta})\|p(\boldsymbol{\theta}))}_{\text{Complexity evaluation}}, \tag{7}$$

which can be decomposed to offer data misfit with model complexity evaluation [29]. It is treated as *hard knowledge* and the ultimate loss is constituted as illustrated in Figure 1. Note that the KL-divergence $D_{\mathrm{KL}}(q_\phi(\boldsymbol{\theta})\|p(\boldsymbol{\theta}))$ is acting as a regularization term in LASSO problem.

Consider modeling the prior of $\boldsymbol{\theta}$ as a zero-mean Gaussian distribution with variance σ^2, as follows:

$$\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\theta}; 0, \sigma^2). \tag{8}$$

Here, the weight $\boldsymbol{\theta}$ is treated as random variable. To sparsify the posterior distributions over $\boldsymbol{\theta}$, an additional set of random variables $\boldsymbol{\gamma}$ is introduced. In this paper, we consider to impose the same scale variable γ_i for the i-th group parameters θ_i to capture the structural dependencies. All the model parameters $\boldsymbol{\theta} = \{\theta_{ij}, 0 < i <= M, 0 < j <= N\}$ is divided into i groups

according to Section III-C, where M denotes the input dimension, N indicates the output. The i-th group parameters θ_i, containing j weights, whose scales appear coupled spatially. The generation of $\boldsymbol{\theta}$ is governed by a hyper-parameter $\boldsymbol{\gamma}$ based on *Hierarchical Hyper-prior distribution*, as follows:

$$\boldsymbol{\theta}|\boldsymbol{\gamma} \sim p(\boldsymbol{\theta}|\boldsymbol{\gamma}), \quad \boldsymbol{\gamma} \sim p(\boldsymbol{\gamma}). \tag{9}$$

By employing sparsity inducing priors for hidden units instead of individual weights, we can prune neurons to avoid complicated and inefficient coding schemes.

It has been proved that imposing *Jeffrey's non-informative prior* for $p(\gamma_i)$ can enforce high sparsity, and it gets simple update rules meanwhile provides accurate estimation [30]. Thus, we choose it as prior of γ_i to sparsify neural networks

$$p(\gamma_i) \propto \frac{1}{|\gamma_i|}, \quad p(\theta_i) \propto \int_0^\infty p(\theta_{ij}|\gamma_i)p(\gamma_i)d\gamma_i = \frac{1}{|\theta_{ij}|} \tag{10}$$

The posterior approximation of $\boldsymbol{\theta}$ is parameterized as a multivariate Gaussian:

$$\begin{aligned} q_\phi(\boldsymbol{\gamma}) &= \prod_{i,j} \mathcal{N}(\gamma_i; \mu_{\gamma_i}, \sigma_{\gamma_i}^2) \Rightarrow q_\phi(\boldsymbol{\theta}) \\ &= \prod_{i,j} \mathcal{N}(\theta_{ij}; \gamma_i\mu_{ij}, \gamma_i^2\sigma_{ij}^2) \end{aligned} \tag{11}$$

Our approach can learn the sparseness of the student network and guide sparsity during the learning process. It exploits the uncertainty of weight by imposing a Bayesian group-sparse-inducing prior on neurons or filters. Specifically, the prior-posterior distance loss is reduced by training, encouraging the posterior distribution of weights to approximate the sparse prior. The posterior is ultimately sparsely distributed.

In the inference stage, we transform our final network to the deterministic version. We utilize the standard convolutional and full-connected layers to replace the corresponding Bayesian hidden layers. By performing variational inference, we can finally obtain the negative KL-divergence from the approximate posterior $q_\phi(\boldsymbol{\gamma})$ to the sparse prior. And the implied group dropout rate is calculated by [18]

$$\alpha_i = \sigma_{\gamma_i}^2/\mu_{\gamma_i}^2 \tag{12}$$

and we can get $\log\alpha_i = (\log\sigma_{\gamma_i}^2 - \log\mu_{\gamma_i}^2) \geq T, T$ is threshold for group pruning. Then we can prune all the neurons under the soft thresholds. A binary mask with sparse columns or rows $\boldsymbol{m}$ is obtained. So the final weights in student model is determined by the masked variational posterior mean [33]

$$\hat{\boldsymbol{\theta}} = diag(\boldsymbol{m} \odot \mu_{\boldsymbol{\gamma}})\boldsymbol{\theta}. \tag{13}$$

The process is illustrated in Figure 3. This group-level sparsification is a natural generalization of the traditional sparse modeling methods. It can effectively model the structural properties of the parameter matrices by clustering together relevant weights, which may belong to the same neurons or filters. It leads to higher performance in pruning out irrelevant coupled weights compared to independent weight pruning.

IV. EXPERIMENTS

To validate the generality and effectiveness of our algorithm for model compression, we carry out extensive experiments with comparisons to the state-of-the-arts methods. We employ the well-known two datasets, including MNIST and CIFAR-10, to evaluate the model size and accuracy based on our VBGS-KD.

1) **MNIST** The MNIST [31] dataset consists of 28 × 28 pixels grey-scale handwritten digits images of 10 classes. We use the standard 60,000 training images and 10,000 test images for experiments. For MNIST, we construct a simple multilayer perceptron (MLP), LeNet-300-100, with two hidden layers of size 300 and 100, and a simple convolutional network, LeNet-5. They have been trained for 200 epochs in all the related experiments. The batch size is 256.
2) **CIFAR-10** The CIFAR-10 [32] dataset consists of 10 categories objects, containing 3-channel color images of 32 × 32 pixels. The dataset is divided into 50,000 train and 10,000 test images. The test part includes exactly 1,000 randomly-selected images from each class. For CIFAR-10, we apply our method with VGG networks [2], that is VGG-11 and VGG-16. A batch size of 128 has been used. The architectures have been trained for 300 epochs.

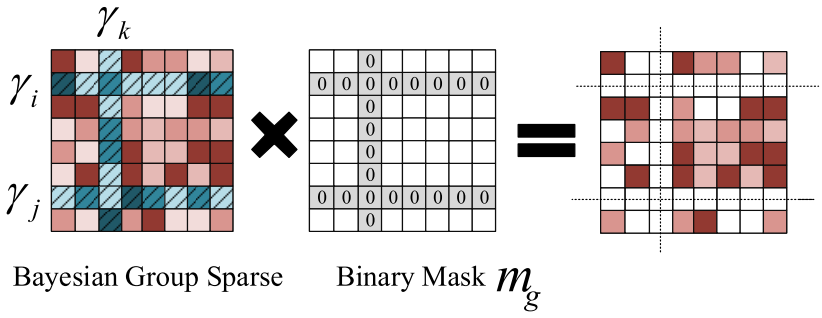


FIGURE 3. Mask generation in student model. Lighter color indicates smaller value. $gamma_i$ control the sparsity of matrix columns or rows. If the dropout rate of $gamma_i$ is larger the threshold, the corresponding neurons or filters can be removed. m_g is group sparse mask. Multiplying by mask is equivalent to prune.

All the datasets are pre-processed using built-in data loading and augmentation in PyTorch. Baseline experimental results for all datasets and corresponding network structures are presented in Table. 1. All the following experiments were run on a computer with a specification of NVIDIA GeForce GTX 1080 Ti GPUs.

TABLE 1. Benchmark results. Here T denotes teacher model, S denotes student model, $K = 10^3$, $M = 10^6$.

Dataset	Role in KD	Model	Acc	#Param
MNIST	T	LeNet-300-100	98.37%	266K
	S	LeNet-5	99.38 %	431K
CIFAR-10	T	VGG-11	90.33%	9.23M
	S	VGG-16	92.16 %	14.72M

A. TRAINING DETAILS

Training details are described as follows. All the experiments used the SGD optimizer with momentum=0.5. A learning rate of 0.003 and were run on the PyTorch framework. The same procedure was used to train the final output architecture. Generally, for the first-pass training model, the learning rate of 0.1 was used, whereas for those trained from pre-trained models and a learning rate of 0.001 was used. The specific hyperparameter configuration was slightly adjusted according to the experimental results.

For initialization, we utilized the pre-trained model. We used the parameters of the pre-training model as the mean of parameters in Bayesian layers, and the variance of Bayesian layers were normal randomly initialized. And we followed the configuration of the standard deviation constraints [33], which can effectively avoid bad local optima for the variational objective.

For inference, we transform our final network to the deterministic version as Section III-C illustrated. Specifically, we use the standard convolutional and linear layers to replace the corresponding Bayesian hidden layers by using the masked variational posterior mean of weight and bias in the original hidden layer as the determined parameters, i.e. the parameter single point estimation.

B. THRESHOLD DETERMINATION

After learning the structural sparseness of weights, we can prune unimportant groups of parameters. Bayesian approximation will give a stochastic presentation, so the weights will not go to zero exactly. Therefore, we first determine the threshold and remove weights meet the dropout condition for inference.

It is worth noting that the threshold for pruning in our method is determined during the training process, which is reasonable, and there is no need for fine-tune. We calculate $\log \alpha_i$ for each group as Section III-C demonstrated, and manually determine the threshold for pruning by visualizing the α-dominated mask value. From Figure 4, we can easily observe that the distribution of $\log \alpha_i$ is separated into two clusters gradually during training. Signals (useful weight groups) are indicated by the cluster of smaller values, whereas noises (useless weight groups) are presented by another larger valued cluster, which can be pruned after training. The total number of each cluster in the distribution represents the number of corresponding useful or useless neurons.

As illustrated in Figure 4, with the training progress, the division of the signal cluster and the noise cluster becomes more and more obvious. Training makes the distribution of signal more concentrated, helping to distinguish useful weights. At the same time, the total number of signal groups is gradually reduced. This means that the number of useful neurons is reduced and the sparseness of the network is augmented. Then, we determine threshold to preserve the signal. We find the threshold is not a sensitive hyperparameter. Generally, we put the threshold at a position where two clusters can be well separated.

C. SPARSITY ANALYSIS

After determining the drop threshold, we can remove the parameter group, whose corresponding value of $\log \alpha_i$ is above the threshold, to obtain the final compressed model.

Table 2 shows the layerwise analysis for student model LeNet-300-100, the teacher model is LeNet-5. It can be observed that both parameter amount and computational complexity are reduced dramatically, that is, the proposed sparsification can effectively compress parameters and accelerate model inference. The sparsity of weight matrices is visualized in Figure 5. Each column represents a neuron of the input layer and rows represent a neuron of the output layer. It can be seen that the matrices appear row sparsity and column sparsity, which corresponds to neurons or filter pruned. Therefore, as the training process progresses, the weights of the network tends to be sparse.

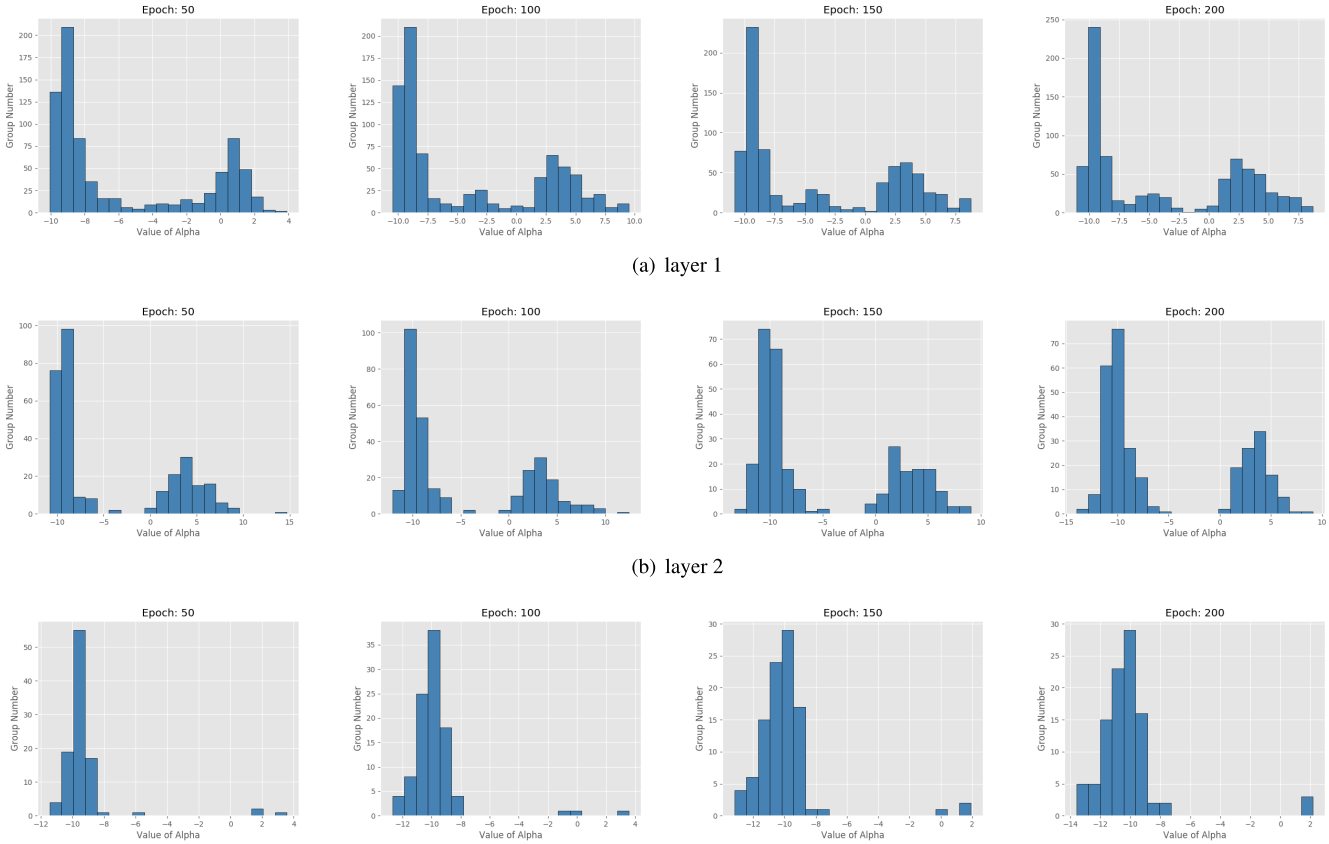
TABLE 2. Experimental results on Lenet-300-100. The model is distilled from LeNet-5. (%) represents the percentage of the compressed parameters (or FLOPs) to the original parameters (or FLOPs).

Layer	Weights	FLOPs	Weights(%)	FLOPs(%)
fc1	235K	470K	19%	19%
fc2	30K	60K	23%	23%
fc3	1K	2K	56%	56%
Total	266K	532K	20%	20%

D. ABALATION STUDY AND ALGORITHM COMPARISON

Our approach takes advantage of both knowledge distillation and model sparsification and pruning. We first make alation study from two respectives, and show the superiority of our method. The experimental results are shown in the first two rows and the last row of Table 3. We compare the method of standard knowledge distillation and the method of straight magnitude pruning.

Our method has several apparent advantages both in terms of model accuracy and compression rate. Compared with the standard knowledge distillation method, the proposed method obtains comparable accuracy and a very small model size. This is because our method can remove the redundant parameters of the student model in knowledge distillation,

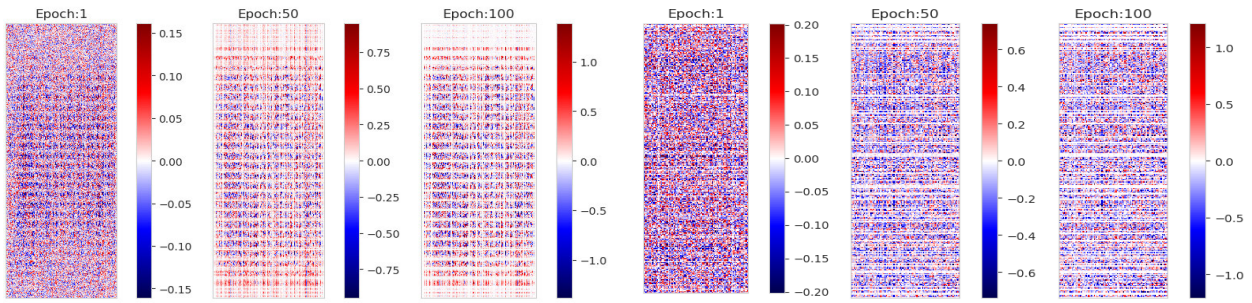


(a) layer 1

(b) layer 2

(c) layer 3

FIGURE 4. Distribution of the thresholds (logα) during training in MNIST task. It can be separated into two clusters, that is signal and noise.



(a) Linear layer 1

(b) Linear layer 2

FIGURE 5. Visualization of group sparsity in the weight matrices of LeNet-300-100 model. Each column represents a neuron of the input layer and rows represent a neuron of the output layer. White color represents the pruned weights, red and blue represent positive and negative weights respectively. The darker the color, the greater the absolute value. In order to better show sparsity of the student model during the training process, the initialization strategy is not used. Fixed pruning threshold is applied here. As the training process progresses, the weights of the network tends to be sparse.

and at the same time alleviate the damage of the model by searching the optimal structure of the model during training. The magnitude pruning method straightly remove the weight smaller than given thresholds, this method can compress the model in a simple way, but it has great influence on the accuracy. Compared with this method, we use knowledge distillation to utilize a large-scale teacher network with better performance to guide the pruning of the model, thus maintain a good model accuracy. Furthermore, the pruning thresholds of our method are inferred in training, hence there is no need to fine-tune.

We also compare our method with two related categories compression methods including: (1) knowledge distillation method with common used regularization, here we consider L1 and group L1 regularizations, and (2) model pruning and sparsification method based on Bayesian Learning, here we

TABLE 3. Comparison of related compression method on LeNet architectures. Here, LeNet-5 is the original model to be compressed. All knowledge distillation experiments use LeNet-300-100 as the student network and LeNet-5 as the teacher network. Our method shows better accuracy under comparable sparsity.

Model	Best Acc.	#Param (Acc.)	Layer Sparsity	Total Sparsity ($\frac{\lvert w=0\rvert}{\lvert w\rvert}$)	Architecture
Standard KD [21]	98.93%	266.2K (98.93%)	0-0-0	0	235200-30000-1000
Magnitude Pruning	98.44%	144.9K (98.30%)	0.48-0.30-0.22	0.46	112005-9044-218
Standard KD+L1	98.68%	57.8K (98.81%)	0.80-0.66-0.16	0.78	27700-10584-845
Standard KD +Group L1	98.95%	170.0K (98.87%)	0.39-0.17-0.13	0.36	144258-19913-870
Han's Pruning [5]	98.36%	21.8K (98.41%)	0.92- 0.91-0.74	0.92	–
SparseVD [18]	98.56%	3.9K (98.08%)	0.99-0.97-0.62	0.98	–
BCGNJ [33]	98.40%	28.7K (98.03%)	0.88-0.95-0.87	0.89	27244-1274-130
Ours	99.04%	52.4K(98.37%)	0.81-0.77-0.44	0.80	44875-7000-560

Original network architecture of LeNet-300-100 is 235200-30000-1000.

consider Han's pruning [5], SparseVD [18], BC-GNJ [33]. Our method achieves a relatively high accuracy, while its compression ratio is comparable to other methods.

Compared with the first category method, although the Standard L1 method achieves a good compression rate, but this kind of method belongs to unstructured compression, and actually needs to calculate the parameter of zero value, so it is actually difficult to reduce the FLOPs and cannot speed up the inference. Our method structurally sparsify the weight matrices as the Standard Group L1 method, which can significantly reduce computational complexity. However, the group L1 method only provides parameter point estimation. Although the accuracy is maintained, the compression rate is not high. Our method can provide uncertainty estimation, thus achieving a more extreme compression performance.

Compared with the second category method, our method achieves close compression rate and preferable accuracy rate. Among them, Han's Pruning and SparseVD are unstructured and sparse methods, which also suffer from high complexity. Compared with the BCGNJ method, we both use Bayesian structural sparsification, but our method utilizes the knowledge from teacher models, which ultimately leads to a relatively high accuracy and a comparable compression performance.

Finally, we evaluate the proposed method on another dataset, CIFAR-10. The experimental results are demonstrated in Table 4, including the model accuracy and corresponding sparsity after pruning. The performance of our model is almost unaffected in most circumstances, so this method is free from fine-tuning. What's more, since our method achieves group-level sparsity, our method is practical to implemented in real-world application. And our compression method can be further boosted by utilizing the bits-back argument [29].

TABLE 4. Extensive experimental results of our method on CIFAR-10 datasets.

	Model	Acc(%)	Sparsity
MNIST	LeNet-300-100	98.37%	0.80
CIFAR-10	VGG-11	88.36%	0.75

V. CONCLUSIONS

In this paper, we present Variational Bayesian Group-level Sparsification for knowledge distillation (VBGS-KD) to incorporate knowledge distillation and group-level sparsificaiton. It utilizes the converge acceleration and accuracy promotion with knowledge transfer, and exploits the structural sparseness in student networks by imposing sparsity-inducing prior and performing variational inference. It results in extremely small-size models. Experimental results show that our method can eliminate the shortcomings of traditional KD routines and can easily produce a high compression rate whereas preserving high accuracy. It's effective in practice compared with the former approaches, and applicable in extensive tasks.

We compress the student model for knowledge distillation by studying the sparseness and the compression performance can be further boost with other compression techniques, such as quantization and coding. To a certain extent, our work shows that the structure of student networks still has a lot to explore. In future work, we will research on the structural design of student-teacher networks and the internal relationship between them.

ACKNOWLEDGMENT

The work presented in this paper was partly supported by Beijing Natural Science Foundation of China (Grant No. L182033), and the Fundamental Research Funds for the Central Universities (2019PTB-001). *(Hao Fu is Co-First author.)*

REFERENCES

[1] Y. Liu, A. Jourabloo, and X. Liu, "Learning deep models for face anti-spoofing: Binary or auxiliary supervision," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 389–398.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556

[3] Z. Lu, V. Sindhwani, and T. N. Sainath, "Learning compact recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2016, pp. 5960–5964.

[4] H. Li, S. De, Z. Xu, C. Studer, H. Samet, and T. Goldstein, "Training quantized nets: A deeper understanding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5811–5821.

[5] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 1135–1143.

[6] A. Polino, R. Pascanu, and D. Alistarh, "Model compression via distillation and quantization," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–8.
[7] S. H. Lee, D. H. Kim, and B. C. Song, "Self-supervised knowledge distillation using singular value decomposition," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2018, pp. 335–350.
[8] Y. Lee, N. Ahn, J. H. Heo, S. Y. Jo, and S.-J. Kang, "Teaching where to see: Knowledge distillation-based attentive information transfer in vehicle maker classification," *IEEE Access*, vol. 7, pp. 86412–86420, 2019.
[9] J. Yim, D. Joo, J. Bae, and J. Kim, "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1–8.
[10] V. Sindhwani, T. Sainath, and S. Kumar, "Structured transforms for small-footprint deep learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 3088–3096.
[11] E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus, "Exploiting linear structure within convolutional networks for efficient evaluation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1269–1277.
[12] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," 2015, *arXiv:1510.00149*. [Online]. Available: http://arxiv.org/abs/1510.00149
[13] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *J. Mach. Learn. Res.*, vol. 5, no. 2, pp. 187–205, 2017.
[14] M. H. Zhu and S. Gupta, "To prune, or not to prune: Exploring the efficacy of pruning for model compression," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 187–205.
[15] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. Peter Graf, "Pruning filters for efficient ConvNets," 2016, *arXiv:1608.08710*. [Online]. Available: http://arxiv.org/abs/1608.08710
[16] R. Yu, A. Li, C.-F. Chen, J.-H. Lai, V. I. Morariu, X. Han, M. Gao, C.-Y. Lin, and L. S. Davis, "NISP: Pruning networks using neuron importance score propagation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9194–9203.
[17] D. Yu, F. Seide, G. Li, and L. Deng, "Exploiting sparseness in deep neural networks for large vocabulary speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2012, pp. 4409–4412.
[18] D. Molchanov, A. Ashukha, and D. P. Vetrov, "Variational dropout sparsifies deep neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2489–2507.
[19] V. Lebedev and V. Lempitsky, "Fast ConvNets using group-wise brain damage," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2554–2564.
[20] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*. [Online]. Available: https://arxiv.org/abs/1503.02531
[21] T. Guo, C. Xu, S. He, B. Shi, C. Xu, and D. Tao, "Robust student network learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 7, no. 2, pp. 1–14, Jul. 2020.
[22] J. Li, K. Fu, S. Zhao, and S. Ge, "Spatiotemporal knowledge distillation for efficient estimation of aerial video saliency," *IEEE Trans. Image Process.*, vol. 29, no. 2, pp. 1902–1914, Nov. 2020.
[23] B. Zhang, D. Xiong, J. Su, and J. Luo, "Future-aware knowledge distillation for neural machine translation," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 27, no. 12, pp. 2278–2287, Dec. 2019.
[24] P. Yang, F. Zhang, and G. Yang, "A fast scene text detector using knowledge distillation," *IEEE Access*, vol. 7, pp. 22588–22598, 2019.
[25] L. Theis, I. Korshunova, A. Tejani, and F. Huszár, "Faster gaze prediction with dense networks and Fisher pruning," in *Proc. Comput. Vis. Pattern Recognit.*, 2018, pp. 2554–2564.
[26] A. Ashok, N. Rhinehart, F. Beainy, and K. M. Kitani, "N2n learning: Network to network compression via policy gradient reinforcement learning," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 2554–2564.
[27] X. Zhu, X. Lan, and S. Gong, "Knowledge distillation by on-the-fly native ensemble," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 7528–7538.
[28] J. Wang, W. Wang, and W. Gao, "Beyond knowledge distillation: Collaborative learning for bidirectional model assistance," *IEEE Access*, vol. 6, pp. 39490–39500, 2018.
[29] G. E. Hinton and D. van Camp, "Keeping the neural networks simple by minimizing the description length of the weights," in *Proc. 6th Annu. Conf. Comput. Learn. Theory (COLT)*, 1993, pp. 5–13.
[30] S. D. Babacan, S. Nakajima, and M. N. Do, "Bayesian group-sparse modeling and variational inference," *IEEE Trans. Signal Process.*, vol. 62, no. 11, pp. 2906–2921, Jun. 2014.
[31] Y. LeCun. (1998). *The MNIST Database of Handwritten Digits*. [Online]. Available: http://yann.lecun.om/exdb/mnist/
[32] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.S. thesis, Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, 2009.
[33] C. Louizos, K. Ullrich, and M. Welling, "Bayesian compression for deep learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3288–3298.
[34] J. Wang, H. Zhang, J. Wang, Y. Pu, and N. R. Pal, "Feature selection using a neural network with group lasso regularization and controlled redundancy," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, May 11, 2020, doi: 10.1109/TNNLS.2020.2980383.

YUE MING (Member, IEEE) received the B.S. degree in communication engineering, the M.Sc. degree in human–computer interaction engineering, and the Ph.D. degree in signal and information processing from Beijing Jiaotong University, China, in 2006, 2008, and 2013, respectively. She worked as a Visiting Scholar at Carnegie Mellon University, USA, from 2010 to 2011. Since 2013, she has been working as a Faculty Member with the Beijing University of Posts and Telecommunications. Her research interests include the areas of biometrics, computer vision, computer graphics, information retrieval, and pattern recognition.

HAO FU received the B.S. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2013, where she is currently pursuing the M.S. degree with the Department of Electronic Engineering. Her research interest includes deep learning in general, with specific interests in model compression and transfer learning.

YIBO JIANG received the B.E. degree in information engineering and the M.E. degree in communications and information system from Zhejiang University, China, in 1997 and 2000, respectively, and the Ph.D. degree in EE from the University of Illinois at Urbana-Champaign, USA, in 2005. From July 2005 to February 2018, he worked at the CR&D Division, Qualcomm Inc. His research interests include the areas of communications, signal processing, information theory, computer vision, and machine learning.

HUI YU (Senior Member, IEEE) is currently a Professor at the University of Portsmouth, U.K. He used to work at the University of Glasgow before moving to the University of Portsmouth. His research interests include methods and practical development in vision, machine learning, and AI with applications to human–machine interaction, virtual and augmented reality, robotics, and geometric processing of facial expression. He serves as an Associate Editor for the IEEE TRANSACTIONS ON HUMAN-MACHINE SYSTEMS and *Neurocomputing* journal.